The Claims

1. (Previously presented) A software architecture for a distributed computing system comprising:

an application configured to handle requests submitted by remote devices over a network; and

an application program interface to present functions used by the application to access network and computing resources of the distributed computing system, wherein the application program interface comprises a set of classes that make available standards-based support for processing XML documents, wherein the set of classes are grouped in the application program interface into a plurality of namespaces, and wherein a first of the plurality of namespaces contains classes and enumerations to support XSLT (Extensible Stylesheet Language Transformations) and a second of the plurality of namespaces contains an XPath parser and evaluation engine.

2. (Original) A software architecture as recited in claim 1, wherein the set of classes comprises:

an XmlReader class that enables non-cached forward only access to XML data;

an XPathNavigator class that enables read-only random access to a data store;

an XslTransform class that enables transforming of XML data using an XSLT stylesheet;

a plurality of Xml Schema classes that enable constructing and editing of schemas;

an XmlResolver class that enables resolving of external XML resources named by a Uniform Resource Identifier (URI);

an XmlDataDocument class that enables structured data to be stored, retrieved, and manipulated through a relational dataset; and

an XmlWriter class that enables a non-cached forward only way of generating streams and files containing XML data.

3. (Original) A software architecture as recited in claim 2, wherein the set of classes further comprises:

an XmlValidatingReader class that enables DTD, XDR and XSD schema validation.

4. (Original) An XmlReader class of an application program interface, embodied on one or more computer readable media that enables non-cached forward only access to XML data, the XmlReader class comprising:

an XmlReader constructor that enables initialization of a new instance of the XmlReader class; and

a Read method that enables reading of nodes of the XML data via the XmlReader class instance.

5. (Original) An XmlReader class of an application program interface as recited in claim 4, wherein the XmlReader class further comprises:

a BaseURI property that identifies a base URI of a current node of the XML data; and

a NodeType property that identifies the type of the current node.

6. (Original) An XPathNavigator class of an application program interface, embodied on one or more computer readable media, that enables read-only random access to a data store, the XPathNavigator class comprising:

an XPathNavigator constructor that enables initialization of a new instance of the XPathNavigator class;

a MoveToFirst method that enables moving to a first sibling of a current node of XML data;

a MoveToNext method that enables moving to a next sibling of the current node;

a MoveToPrevious method that enables moving to a previous sibling of the current node;

a MoveToFirstChild method that enables moving to a first child of the current node;

a MoveToParent method that enables moving to a parent of the current node; and

a NodeType property that enables obtaining the type of the node that is moved to.

7. (Original) An XslTransform class of an application program interface, embodied on one or more computer readable media, that enables transforming of XML data using an XSLT stylesheet, the XslTransform class comprising:

an XslTransform constructor that enables initialization of a new instance of the XslTransform class;

a Load method that enables loading of the XSLT stylesheet; and

a Transform method that enables transforming of the specified XML data using the loaded XSLT stylesheet and outputs the results.

8. (Original) A set of XmlSchema classes of an application program interface, embodied on one or more computer readable media, that enable constructing and editing of schemas, the set of XmlSchema classes comprising:

a Schema class that contains a definition of a schema;

a SchemaObject class that enables creating of an empty schema; and

a SchemaCollection class that contains a cache of defined XML Schema Definition language (XSD) and XML-Data Reduced Language (XDR) schemas.

9. (Original) An XmlResolver class of an application program interface, embodied on one or more computer readable media, that enables resolving of external XML resources named by a Uniform Resource Identifier (URI), the XmlResolver class comprising:

a ResolveURI method that enables resolving the absolute URI from a base URI and a relative URI; and

a GetEntity method that enables mapping of the resolved URI to an object containing identified resource.

10. (Original) An XmlDataDocument class of an application program interface, embodied on one or more computer readable media, that enables structured data to be stored, retrieved, and manipulated through a relational dataset, the XmlDataDocument class comprising:

a DataSet property that enables obtaining of a dataset that provides a relational representation of the data in a document;

a Load method that enables loading of the document using a specified data source and synchronizing the dataset with the loaded data.

11. (Original) An XmlWriter class of an application program interface, embodied on one or more computer readable media, that enables a non-cached forward only way of generating streams and files containing XML data, the XmlWriter class comprising:

an XmlWriter constructor that enables initialization of a new instance of the XmlWriter class; and

an WriteState property that enables obtaining of the state of an instance of the XmlWriter class; and

a plurality of Write methods that enable writing XML data via the instance of the XmlWriter class.

12. (Original) An XmlValidatingReader class of an application program interface, embodied on one or more computer readable media, that enables DTD, XDR and XSD schema validation, the XmlValidatingReader class comprising:

a ValidationType property that enables obtaining an indication of what type of validation to perform on a document;

a Read method that enables reading of nodes of the document so that validation of the document can be performed.

13. (Original) An XmlValidatingReader class of an application program interface as recited in claim 12, where the type of validation to perform is validation according to DTD.

14. (Original) An XmlValidatingReader class of an application program interface as recited in claim 12, where the type of validation to perform is validation according to XSD schemas.

15. (Original) An XmlValidatingReader class of an application program interface as recited in claim 12, where the type of validation to perform is validation according to XDR schemas.

16. (Previously presented) A distributed computer software architecture, comprising:

one or more applications configured to be executed on one or more computing devices, the applications handling requests submitted from remote computing devices;

a networking platform to support the one or more applications; and

an application programming interface to interface the one or more applications with the networking platform, wherein the application program interface comprises a set of classes that make available standards-based support for processing documents written in a markup language, wherein the set of classes are grouped in the application programming interface into a plurality of namespaces, and wherein a first of the plurality of namespaces contains classes and enumerations to support XSLT (Extensible Stylesheet Language Transformations), a second of the plurality of namespaces contains an XPath parser and evaluation engine, and a third of the plurality of namespaces contains classes used to serialize objects into XML format documents or streams.

17. (Original) A software architecture as recited in claim 16, wherein the set of classes comprises:

an XmlReader class that enables non-cached forward only access to XML data;

an XPathNavigator class that enables read-only random access to a data store;

an XslTransform class that enables transforming of XML data using an XSLT stylesheet;

a plurality of Xml Schema classes that enable constructing and editing of schemas;

an XmlResolver class that enables resolving of external XML resources named by a Uniform Resource Identifier (URI);

an XmlDataDocument class that enables structured data to be stored, retrieved, and manipulated through a relational dataset; and

an XmlWriter class that enables a non-cached forward only way of generating streams and files containing XML data.

18. (Original) A software architecture as recited in claim 17, wherein the set of classes further comprises:

an XmlValidatingReader class that enables DTD, XDR and XSD schema validation.

19. (Previously presented) A computer system including one or more microprocessors and one or more software programs, the one or more software programs utilizing an application program interface to request services from an operating system, the application program interface including separate commands to request services that make available support for processing XML documents, the separate commands being grouped into different namespaces including a first namespace to support XSLT (Extensible Stylesheet Language Transformations) and a second namespace to serialize objects into XML format documents or streams.

20. (Original) A software architecture as recited in claim 19, wherein the commands include:

non-cached forward only access to XML data;

read-only random access to a data store;

transforming of XML data using an XSLT stylesheet;

constructing and editing of schemas;

resolving of external XML resources named by a Uniform Resource Identifier (URI);

storage, retrieval, and manipulation of structured data through a relational dataset; and

non-cached forward only generation of streams and files containing XML data.

21. (Original) A software architecture as recited in claim 20, wherein the commands further include:

validating DTD, XDR and XSD schemas.

22. (Previously presented) A method comprising:

receiving one or more calls from one or more remote devices over a network, wherein the one or more calls are to one or more functions that make available support for processing XML documents, the one or more functions being grouped into a plurality of namespaces with a first namespace containing an XPath parser and evaluation engine and a second namespace containing classes used to serialize objects into XML format documents or streams; and

performing the requested XML document processing.

23. (Original) A software architecture as recited in claim 22, wherein the one or more functions comprises:

an XmlReader function that enables non-cached forward only access to XML data;

an XPathNavigator function that enables read-only random access to a data store;

an XslTransform function that enables transforming of XML data using an XSLT stylesheet;

a plurality of Xml Schema functions that enable constructing and editing of schemas;

an XmlResolver function that enables resolving of external XML resources named by a Uniform Resource Identifier (URI);

an XmlDataDocument function that enables structured data to be stored, retrieved, and manipulated through a relational dataset; and

an XmlWriter function that enables a non-cached forward only way of generating streams and files containing XML data.

24. (Original) A software architecture as recited in claim 23, wherein the one or more functions further comprises:

an XmlValidatingReader function that enables DTD, XDR and XSD schema validation.

25. (Previously presented) A method comprising:

calling, to one or more remote devices over a network, one or more functions that make available support for processing XML documents, the one or more functions being grouped into a plurality of namespaces with a first namespace containing classes and enumerations to support XSLT (Extensible Stylesheet Language Transformations) and a second namespace containing classes used to serialize objects into XML format documents or streams;

receiving, from the one or more remote devices, a response to the calling.

26. (Original) A software architecture as recited in claim 25, wherein the one or more functions comprises:

an XmlReader function that enables non-cached forward only access to XML data;

an XPathNavigator function that enables read-only random access to a data store;

an XslTransform function that enables transforming of XML data using an XSLT stylesheet;

a plurality of Xml Schema functions that enable constructing and editing of schemas;

an XmlResolver function that enables resolving of external XML resources named by a Uniform Resource Identifier (URI);

an XmlDataDocument function that enables structured data to be stored, retrieved, and manipulated through a relational dataset; and

an XmlWriter function that enables a non-cached forward only way of generating streams and files containing XML data.

27. (Original) A software architecture as recited in claim 26, wherein the one or more functions further comprises:

an XmlValidatingReader function that enables DTD, XDR and XSD schema validation.